In the Specification

Please replace paragraph [0035] on pg. 11 with the following replacement paragraph with changes. No new matter has been added.

[0035] Figure 3 is a flow chart of an illustrative process for providing remote computer access. As shown, at step 310, a request to initiate a remote access session is received at one of the server computing systems 112, for example system 112a, from one of the client computing systems 116, for example system 116a, via network 114. In an illustrative embodiment, the request is received and handled by the remote access server 214a. At step 312, remote access server 214a causes the desired application, which may be application 212a of Fig. 2, to be launched on system 112a. During execution of application 212a on OS 210, remote access server 214a recognizes, at step 314, an output-related instruction such as, for example, an instruction related to displaying data or generating a sound. At step 316, translator 216a translates the instruction from a native format for execution by OS 210a into an open system data item such as, for example, an XML element. XML is a markup language that is used to describe data and is known by those skilled in the art. While those skilled in the computing arts are knowledgeable regarding XML and can implement an XML system, background regarding implementing XML systems is disclosed at www.w3c.org, the contents of which are hereby incorporated by reference in their entirety. Generally, an XML element is defined by at least a beginning tag, a data item corresponding to the element content, and a closing tag. For example, a native OS 210 instruction for displaying a file (referred to as *filename*) may be translated into an XML element such as the following: <display>filename</display>. The tag "<display>" represents the beginning of the XML element, the tag "</display>" represents the end of the XML element, and *filename* corresponds to the data to be displayed. Similarly, a native OS 210 instruction for playing an audio file may be translated by translator 216 into an XML formatted data item, for example, such as the following:<play>*filename*</*play*>. Indeed, an XML equivalent may be developed for all native instructions. Translator 216 may maintain a database matching native instructions to corresponding XML elements. For example, an instruction to display a file may be matched to XML tags "<display>" and "</display>". Upon receiving an output native instruction, translator 216 may access the database to identify the corresponding

XML element(s). ~~At step 316, OS 210a may also execute the native format instruction that is being translated so that the output is generated at server computing system 112 as well as at client computing system 116.~~

In the claims:

1. (Currently Amended) A method for providing remote computer control of ~~an application executing on~~ a second computer from a first computer over a network, comprising:

receiving a ~~first~~ user input ~~instruction~~ by a first operating system of the first computer via a first computer input peripheral device ~~running on the first computer for execution, the first user input instruction being operationally compatible with a first computer language of the first operating system and operationally incompatible with a second computer language of a second operating system executing on the second computer the first operating system being incompatible with the second operating system thereby requiring the first user input instruction to be translated from the first computer language of the first operating system in order to be executed by the second operating system;~~

at the first computer, translating the first user input ~~instruction from the first computer language of the first operating system~~ into ~~a data script defining~~ at least one XML item ~~utilizing a first device driver resident in the operating system on the first computer, wherein the first device driver formats the first user input instruction into at least one XML item corresponding to the first user input instruction~~; and

transmitting the ~~data script defining the~~ at least one XML item from the first computer to the second computer;

translating the ~~data script defining the~~ at least one XML item into a ~~second~~ user input instruction ~~the second computer language~~ of a second operating system ~~utilizing a second device driver in the second operating system~~ on the second computer that instructs the second operating system to execute an instruction equivalent to receiving the user input at an input peripheral device of the second computer, wherein the second operating system translates the XML item into the user input instruction according to a database that comprises XML items associated to corresponding instructions of the second operating system ~~device driver translates the at least one XML item corresponding to the first user input instruction into the second user input instruction, the second user input instruction being compatible with the second language of the second operating system running on the second computer and incompatible with the first computer language of the first operating system running on the first computer, the second user~~

~~input instruction being functionally similar to the first user input instruction for execution on the second computer; and~~

executing the ~~second~~ user input instruction on the second computer.

2-5. (Cancelled)

6. (Currently Amended) The method of claim 1 ~~5~~, wherein receiving the ~~first~~ user input ~~instruction for inputting data~~ comprises receiving ~~an instruction indicating~~ a mouse input.

7. (Currently Amended) The method of claim 1 ~~5~~, wherein receiving the ~~first~~ user input ~~instruction for inputting data~~ comprises receiving ~~an instruction indicating~~ a keyboard input.

8. (Currently Amended) The method of claim 1, wherein translating ~~the first~~ the ~~first~~ user ~~instruction~~ input into ~~a data script defining~~ the at least one XML item comprises generating a first XML tag defining the beginning of the XML item, generating a data item corresponding to the first user instruction, and generating a second XML tag defining the end of the XML item.

9. (Currently Amended) The method of claim 1, wherein transmitting the at least one XML item ~~data~~ to ~~a~~ the second computer comprises transmitting the data using HTTP.

10. (Currently Amended) The method of claim 1, wherein translating the at least one XML item ~~data~~ into a second input instruction comprises identifying a first XML tag defining the beginning of an XML item, identifying a data item corresponding to a user input instruction, identifying a second XML tag defining the end of an XML item.

11. (Cancelled)

12. (Previously Presented) A computer readable medium having computer-implementable instructions stored thereon for performing the method recited in claim 1.

13-19. (Cancelled)

20. (Currently Amended) A system for remote computer access between computing systems ~~with incompatible operating systems~~, comprising:

a first computing system having stored thereon software which when executed on the first computing system:

receives a user input via a first user interface of the first computing system;

identifies user input instructions generated by a operating system on the first computer system in a first computer language, the user input instructions relating to generating a system output via a second user interface of the first computing system in response to the user input,

translates the user input instructions into a first non-proprietary data script defining an outgoing software object ~~corresponding to the user input instructions~~ such that the outgoing software object provides instructions to the second computer to execute instructions corresponding to the user input received via a first user interface of the first computer, the translation being accomplished by referencing a database to match the user input instructions in the first computer language to contents of the first non-proprietary script ~~a first device driver within the operating system on the first computing system~~,

transmits the outgoing software object to a second computing system, and

receives an incoming software object comprising a second non-proprietary data script from the second computing system reflecting a response to the user input instructions for execution on the second user interface of the first computing system, wherein the second non-proprietary data script is translated ~~by the first device driver~~ into a system output instruction in the first computer language by referencing the database to match contents of the second non-proprietary script to instructions in the first computer language ~~being compatible with the operating system of the first computing system and being incompatible with a second computer language of a second computing system~~, the system output instruction then being executed on the first computing system as a system output via the second user interface.

21. (Currently Amended) A method for providing remote computer access, comprising:

at a first computer, receiving outgoing instructions relating to generating an output on the first computer from a first operating system in a first computer language executing on the first computer~~, the instructions being compatible with the first operating system and incompatible with a second operating system running on a second computer using a second computer language~~;

creating data defining a first XML item corresponding to the outgoing instructions such that the XML item provides instructions to the second computer to execute instructions corresponding to the output generated on the first computer, wherein the outgoing instructions are translated into the first XML item ~~element~~ at the first computer by referencing a database that comprises XML items associated to corresponding instructions of the first computer language;

transmitting the first XML item ~~element~~ from the first computer to the second computer;

at the first computer, receiving data from the second computer defining a second XML item in response to the outgoing instructions;

creating incoming instructions relating to generating the output on the first computer from the data defining the second XML item, wherein the incoming instructions are translated from the second XML item at the first computer by referencing the database ~~and after which the incoming instructions are compatible with the first computer language of the first operating system running on the first computer and are incompatible with the second computer language of the second operating system running on the second computer~~; and

executing the incoming instructions to generate the output at the first computer.

22. (Previously Presented) The method of claim 21, wherein receiving incoming instructions relating to generating output comprises receiving instructions relating to generating visual or audio output.

23. (Previously Presented) The method of claim 21, wherein creating the first XML item corresponding to the outgoing instructions relating to generating output comprises generating at least a first XML tag defining the beginning of the first XML item, generating a data item corresponding to the instruction relating to generating output; and generating at least a second XML tag defining the ending of the first XML item.

24. (Previously Presented) The method of claim 21, wherein transmitting the data defining the first XML item comprises transmitting the data defining the first XML item using HTTP protocol.

25. (Previously Presented) The method of claim 21, wherein creating incoming instructions relating to generating the output comprises identifying a first XML tag identifying the beginning of the XML item, identifying a data item corresponding to an input, and identifying a second XML tag identifying the ending of the XML item.

26. (Currently Amended) A method for providing remote computer access between computing systems ~~with incompatible operating systems~~, comprising:

receiving a first user input instruction relating to a user input received via a first user interface of the first computer by a first operating system on the first computer using a first computer language~~, the first user input instruction being compatible with the first operating system and incompatible with a second operating system on the second computer using a second computer language~~;

creating data defining a first software object in a non-proprietary format corresponding to the first user input instruction relating to the user input, wherein the data defining the first software object is created by referencing a first database comprising software objects in non-proprietary formats associated with user input instructions;

transmitting the first software object from the first computer to the second computer;

at the second computer, translating the first software object from the non-proprietary format to a second user input instruction by referencing a second database comprising software objects in non-proprietary formats associated with user input instructions ~~compatible with the second operating system and incompatible with the first operating system~~;

executing the second user input instruction by the second computer, wherein the second user input instruction corresponds to the first user input received via the first user interface of the first computer;

receiving data from the second operating system related to the second user input instruction being executed, the data defining a first system output instruction, the first system output instruction relating to the first user input instruction ~~and being compatible with the second~~

~~operating system executing on the second computer and incompatible with the first operating system on the first computer;~~

creating data defining a second software object in the non-proprietary format that corresponds to the second user input instruction, wherein the data defining the second software object is created by referencing the second database;

transmitting the second software object from the second computer to the first computer;

at the first computer, translating the second software object to a second system output instruction ~~being compatible with the first operating system and incompatible with the second operating system;~~ and

executing the second system output instruction to render the user output by the first computer on a second user interface.

27. (Previously Presented) The method of claim 25, wherein transmitting the data defining the first and second software objects comprises using the HTTP protocol to transmit the first and second software objects.

28. (Previously Presented) The system of claim 20 wherein the first user interface is different from the second user interface.

29. (Previously Presented) The system of claim 26 wherein the first user interface is different from the second user interface.

30. (Currently Amended) The system for remote computer access ~~between computing systems with incompatible operating systems~~ of claim 20, further comprising the second computing system having stored thereon software which when executed on the second computing system:

receives the outgoing software object from the first computing device;

translates the first non-proprietary data script using a second device driver executing in conjunction with a second operating system executing on the second computer system into the user input instructions identified by the first computing system ~~but operationally compatible with a second operating system executing on the second~~

~~computer system and operationally incompatible with the operating system executing on the first computer system~~;

executes the user input instructions compatible with the second operating system;

identifies system output instructions ~~operationally compatible with a second operating system executing on the second computer system and operationally incompatible with the operating system executing on the first computer system~~, the system output instructions being responsive to the user input instructions identified by the first computing system,

translates the system output instructions into a second non-proprietary data script defining an incoming software object utilizing the second device driver

transmits the incoming software object; and

a communications network operably coupled between the first computing system and the second computing system for transmitting the first and second non-proprietary data scripts defining incoming and outgoing software objects between the first computing system and the second computing system.